KodeMed

Politica di deployment OpenShift

Per DevOps, Platform Engineer e architetti IT

Version 2026.5.15.59502 | KodeMed AG

Architettura — Componenti

Servizi distribuiti in OpenShift

Servizio	Tecnologia	Porta	Tipo
kodemed-server	Java 21, Spring Boot 3.4	8080	Deployment
kodemed-dataserver	Java 21, Spring Boot 3.4	8081	Deployment
kodemed-grouper	Java 21, Spring Boot 3.4	8082	Deployment
kodemed-ui	React 18 / nginx	3000	Deployment
PostgreSQL 16	Esterno (non un pod)	5432	DB esterna

Client Windows (non containerizzati):

- KodeMed.Interface — DLL COM .NET 9.0 (integrazione HIS)
- KodeMed.CodingClient — App system tray .NET 9.0 (bridge WebSocket)

Limiti di risorse per pod

Richieste/limiti CPU e memoria per ogni servizio

Pod	CPU Req	CPU Limit	Mem Req	Mem Limit
kodemed-server	250m	1000m	512Mi	1Gi
kodemed-dataserver	200m	500m	256Mi	1Gi
kodemed-grouper	200m	500m	512Mi	1Gi
kodemed-ui (nginx)	50m	200m	32Mi	64Mi
TOTALE (x2 repliche)	1400m	4400m	2600Mi	6.1Gi

JVM: -XX:+UseContainerSupport -XX:MaxRAMPercentage=75.0 — rispetta automaticamente il memory limit del container.

Alta disponibilità

Minimo 2 repliche per servizio con rolling update

Servizio	Repliche	Rolling Update	Vincoli
kodemed-server	2	maxUnavail: 0, maxSurge: 1	Sticky session obbligatorie
kodemed-dataserver	2	maxUnavail: 1, maxSurge: 1	Stateless — nessun vincolo
kodemed-grouper	2	maxUnavail: 1, maxSurge: 1	Stateless — nessun vincolo
kodemed-ui	2	maxUnavail: 1, maxSurge: 1	Stateless — nessun vincolo

Sticky Sessions — kodemed-server

- Connessioni WebSocket in memoria JVM (ConcurrentHashMap)
- La Route OpenShift DEVE configurare l'affinità di sessione:
- haproxy.router.openshift.io/balance: source
- haproxy.router.openshift.io/timeout: 86400s
- Riavvio pod: client si riconnettono in 60s — sessioni in DB, nessuna perdita

Health Probes

Sonde readiness, liveness e startup per tutti i servizi

Servizio	Endpoint	Porta	Init Delay	Periodo
kodemed-server	/actuator/health	8080	60s	10s
kodemed-dataserver	/actuator/health	8081	60s	10s
kodemed-grouper	/actuator/health	8082	60s	10s
kodemed-ui	/	3000	5s	10s

Tipi di sonde

- Readiness — determina se il pod può ricevere traffico (rimosso dal Service se fallisce)
- Liveness — determina se il pod è vivo (container riavviato se fallisce)
- Startup — protegge app lente all'avvio (JVM warmup), failureThreshold: 12

Database — PostgreSQL esterno

Nessun pod DB consentito in OpenShift — DB esterno con SSL in-transit

Configurazione

- Nessun pod DB in OpenShift — conforme alla politica del cluster
- SSL in-transit: sslmode=require&sslrootcert=/certs/ca.crt
- Certificato SSL montato via Secret OpenShift (kodemed-db-ssl-cert)
- Connection pool: HikariCP max 15 conn, min 5 idle, per servizio

Schema & Migrazioni

- Schema condiviso: Server prefisso km_app_, DataServer prefisso km_data_
- Migrazioni Flyway automatiche all'avvio
- Nessuno script SQL manuale richiesto
- Eseguire sempre backup del database prima dell'aggiornamento

Sicurezza immagini & SCC

Immagini immutabili, scansione vulnerabilità e Security Context Constraints OpenShift

Immagini immutabili

- readOnlyRootFilesystem: true
- allowPrivilegeEscalation: false
- runAsNonRoot: true, runAsUser: 1001
- Nessuna installazione a runtime — config solo via env vars
- Nessun nested container

Scansione & Firme

- Scansione Trivy ad ogni build CI
- CVE critiche = immagine bloccata
- Cosign per integrità supply-chain
- Base: eclipse-temurin:21-jre-alpine

Immagine	UID	SCC
kodemed-server	1001	restricted (OK)
kodemed-dataserver	1001	restricted (OK)
kodemed-grouper	1001	restricted (OK)
kodemed-ui	root*	AZIONE RICHIESTA

Azione: migrare kodemed-ui a nginxinc/nginx-unprivileged o configurare UID > 1000.

Domande aperte — Risposte

Risposte alle domande della riunione

SI

PDB (PodDisruptionBudget)?

minAvailable: 1 per servizio. Con 2 repliche, il cluster può evacuare max 1 pod alla volta.

NO

Operatore Kubernetes?

Non necessario. Risorse K8s standard (Deployment, Service, Route) sufficienti. Nessun CRD custom.

MISTO

Stateless vs Stateful?

DataServer, Grouper, UI = stateless. Server = stateful (WebSocket in JVM). Tutti come Deployment. Sticky session su Route.

HELM

Tipo di deployment?

Helm chart raccomandati. Parametrizzabili per env (values-test.yaml, values-prod.yaml). Rollback facile.

SI

Compatibile GitOps?

Immagini immutabili + tag versionati (YYYY.M.D.BUILD) + config dichiarativa. Workflow ArgoCD sync.

Matrice RASCI

Responsabilità chiaramente definite per ogni attore

Attività	Mieres IT	Cliente Piattaforma	Cliente IT
Build & scan immagini	R	I	I
Pubblicare immagini su JFrog	R	S	I
Creare namespace OpenShift	C	R	A
Configurare quote/limiti	C	R	A
Creare Helm chart	R	C	I
Deployment (test & prod)	S	R	A
Config Routes/Ingress + SSL	C	R	A
Provisioning PostgreSQL + SSL	C	R	A
Gestire Secret OpenShift	C	R	A
Rolling update	R (immagini)	R (deploy)	A
Supporto applicativo L2/L3	R	S	I
Backup & ripristino DB	C	R	A

R = Responsible | A = Accountable | S = Supportive | C = Consulted | I = Informed

Azioni richieste

Azioni da completare prima del deployment

#	Action	Cliente Piattaforma	Prio
1	Creare namespace kodemed-test su OpenShift	Cliente	Alta
2	Provisioning PostgreSQL 16 con SSL attivato	Cliente	Alta
3	Configurare repo Docker in JFrog	Cliente	Alta
4	Configurare repo Helm in JFrog	Cliente	Alta
5	Fornire credenziali JFrog a Mieres IT	Cliente	Alta
6	Adattare kodemed-ui per nginx non-root (SCC)	Mieres IT	Alta
7	Creare Helm chart	Mieres IT	Alta
8	Creare Secret OpenShift (DB, encryption, JWT)	Cliente	Alta
9	Configurare Route WebSocket + sticky session	Cliente	Alta
10	Validare matrice RASCI con tutti gli stakeholder	Cliente IT	Media